INTRODUCTION

The memory board you have received can fill any one of the four 8K blocks of memory in a 32K system. These blocks of memory are illustrated in Figure 1 with their appropriate addresses. If your system is larger than 32K refer to the modification outlined on page 11 of this manual.

FIGURE 1

	8K 0 to 8K	8K 8 to 16K	8K 16 to 24K	8K 24 to 32K
	system ROM	transparent to SINCLAIR system	- - available RAM - -	
	BLOCK 0	BLOCK 1	BLOCK 2	BLOCK 3
A13	0	1	0	1
A14	0	0	1	1

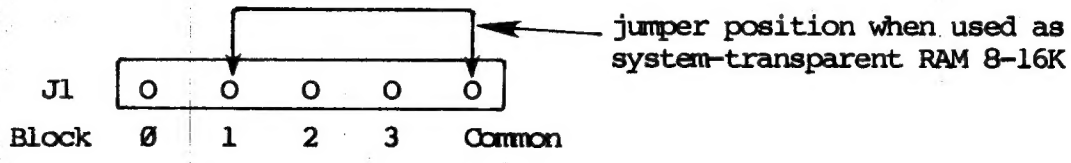
The first 8K is occupied by the operating system and BASIC interpreter of the ZX81 computer. You can modify the SINCLAIR system by using this board in the 0 to 8K slot.

The second 8K is not occupied and is an ideal place to store your own system utilities and machine language routines. It is primarily for this location that this 8K board, with its nonvolatile memory, is designed.

The remaining 16K is available for Sinclair BASIC system RAM. It may be 1K for a simple ZX81, 2K for an upgraded ZX81 or TS1000, or 16K for a system with a 16K RAM pack. Note that if you require a full 16K of system RAM it is more economical to purchase a Sinclair RAM pack rather than use two of these boards. However, this board can be jumper-programmed to occupy either 8K of system RAM.

Selection of the different 8K blocks of memory is achieved by jumper J1 as shown in Figure 2.

FIGURE 2



Each configuration will be described in turn.

Issue 5 March 1984

SYSTEM ROM Ø - 8K

This configuration is used only when you wish to modify the Sinclair operating system. The board can be populated by HM6116LP CMOS RAM with low-power back-up or by 2716 or 2732 EPROM's. Jumper configurations for the different memory devices are discussed in the next section. Note that the character table in ROM is addressed by the Sinclair system logic chip and not by the CPU (the address bus is split). Initially you may wish to retain access to the character table (768Ø to 8191 or 7.5K to 8K). Note that the character table is read during the refresh cycle and that access to the ROM should be allowed when $\overline{RFSH}$ is low (active).

SYSTEM TRANSPARENT RAM 8 - 16K

The block of memory from 8 to 16K is transparent to the Sinclair system -- it doesn't know it's there. This area of memory is therefore an ideal place to store, either permanently or temporarily, machine language routines or data which are to be used by the BASIC system. Some examples of routines you might store in this block of memory are:

* fast graphics routines
* custom mathematical or statistical functions
EP-2X 12-14 * Sinclair code / ASCII conversion tables
* octal/decimal/hex conversion routines
* I/O servicing routines for control applications
* a checksum routine
* a routine for resequencing BASIC programs
* a routine for merging BASIC programs and other toolkit utilities
FD-2X- 10-12 * a disc operating system (DOS) or other development system
* speech synthesis routines
* additional BASIC commands
* EPROM programming routines

Sample routines are described at the end of this manual.

The use of HM6116LP-3 2K CMOS RAM memory IC's with a back-up power supply means that routines stored in the RAM are nonvolatile -- the RAM retains its memory even when the ZX81 is switched off or reset. Moreover, being RAM, the routines stored in the memory are easily modified. The lithium cell supplied with the board should maintain sufficient reserve power for about ten years for a fully populated board. See note on page 8.

Once you have established your system utilities and other machine language routines, you may wish to replace the 6116LP-3 CMOS RAM's by 2716 or 2732 EPROM's or equivalent EEPROM's. This board can accomodate EPROM's in place of the CMOS RAM's. Jumper J2 is used to select the memory device type as shown in Figure 3. A mixture of RAM and EPROM is allowed -- see page 1Ø.

Note that some commercially available accessories for the ZX81 system -- such as printers or disc drives -- allocate some portion of the transparent 8K block to the control of these systems. To accomodate such peripherals, the board can be partially populated. Alternatively, a paging system may be used. In such a system several devices occupy the same region but only one is enabled or selected at a particular time -- see page 13.

SYSTEM ROM Ø - 8K

This configuration is used only when you wish to modify the Sinclair operating system. The board can be populated by HM6116LP CMOS RAM with low-power back-up or by 2716 or 2732 EPROM's. Jumper configurations for the different memory devices are discussed in the next section. Note that the character table in ROM is addressed by the Sinclair system logic chip and not by the CPU (the address bus is split). Initially you may wish to retain access to the character table (768Ø to 8191 or 7.5K to 8K). Note that the character table is read during the refresh cycle and that access to the ROM should be allowed when $\overline{RFSH}$ is low (active).

SYSTEM TRANSPARENT RAM 8 - 16K

The block of memory from 8 to 16K is transparent to the Sinclair system -- it doesn't know it's there. This area of memory is therefore an ideal place to store, either permanently or temporarily, machine language routines or data which are to be used by the BASIC system. Some examples of routines you might store in this block of memory are:

* fast graphics routines
* custom mathematical or statistical functions
EP-2X 12-14 * Sinclair code / ASCII conversion tables
* octal/decimal/hex conversion routines
* I/O servicing routines for control applications
* a checksum routine
* a routine for resequencing BASIC programs
* a routine for merging BASIC programs and other toolkit utilities
FD-2X- 10-12 * a disc operating system (DOS) or other development system
* speech synthesis routines
* additional BASIC commands
* EPROM programming routines

Sample routines are described at the end of this manual.

The use of HM6116LP-3 2K CMOS RAM memory IC's with a back-up power supply means that routines stored in the RAM are nonvolatile -- the RAM retains its memory even when the ZX81 is switched off or reset. Moreover, being RAM, the routines stored in the memory are easily modified. The lithium cell supplied with the board should maintain sufficient reserve power for about ten years for a fully populated board. See note on page 8.

Once you have established your system utilities and other machine language routines, you may wish to replace the 6116LP-3 CMOS RAM's by 2716 or 2732 EPROM's or equivalent EEPROM's. This board can accomodate EPROM's in place of the CMOS RAM's. Jumper J2 is used to select the memory device type as shown in Figure 3. A mixture of RAM and EPROM is allowed -- see page 1Ø.

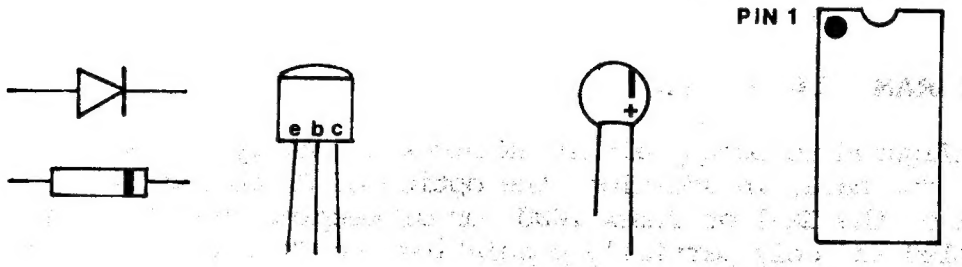
Note that some commercially available accessories for the ZX81 system -- such as printers or disc drives -- allocate some portion of the transparent 8K block to the control of these systems. To accomodate such peripherals, the board can be partially populated. Alternatively, a paging system may be used. In such a system several devices occupy the same region but only one is enabled or selected at a particular time -- see page 13.

COMPONENTS

DO NOT REMOVE THE CMOS RAM IC FROM ITS PROTECTIVE FOAM UNTIL NECESSARY.

Check and identify each component against the parts list. Note the polarity of the diodes, the tantalum capacitor, the transistors, and the IC's as shown in Figure 5. The polarity of the lithium cell is marked on it.

FIGURE 5



diode NPN transistor tantalum capacitor IC (from top)

The printed circuit board is silkscreened with the parts placement diagram shown in Figure 6. All components except two are mounted on the side with the printing and all the soldering is done on the other side marked "circuit side".

The following procedure assumes that the board will be allocated the transparent region 8 - 16K and that 6116LP-3 CMOS RAM's with a back-up power supply will be used. Changes to the procedure for other options are summarized later -- refer to these now if you choose an alternative option. The use of movable jumpers allows you to keep your options open and vary the use to which you put the board.

FIGURE 6

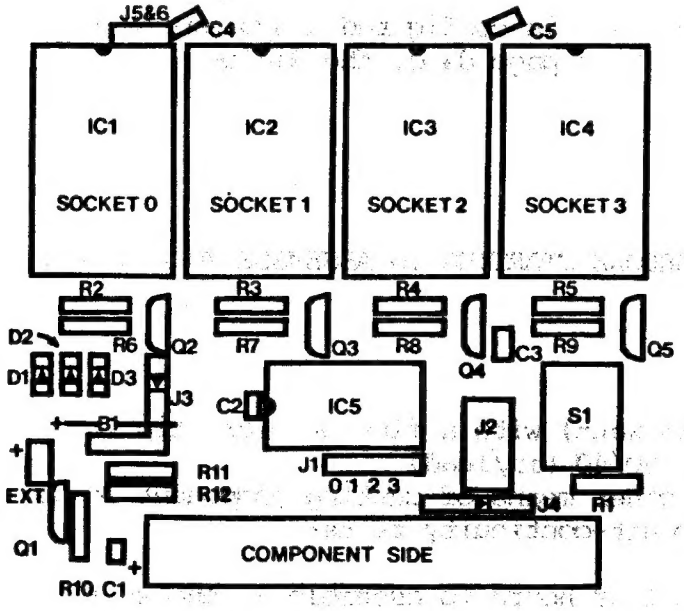
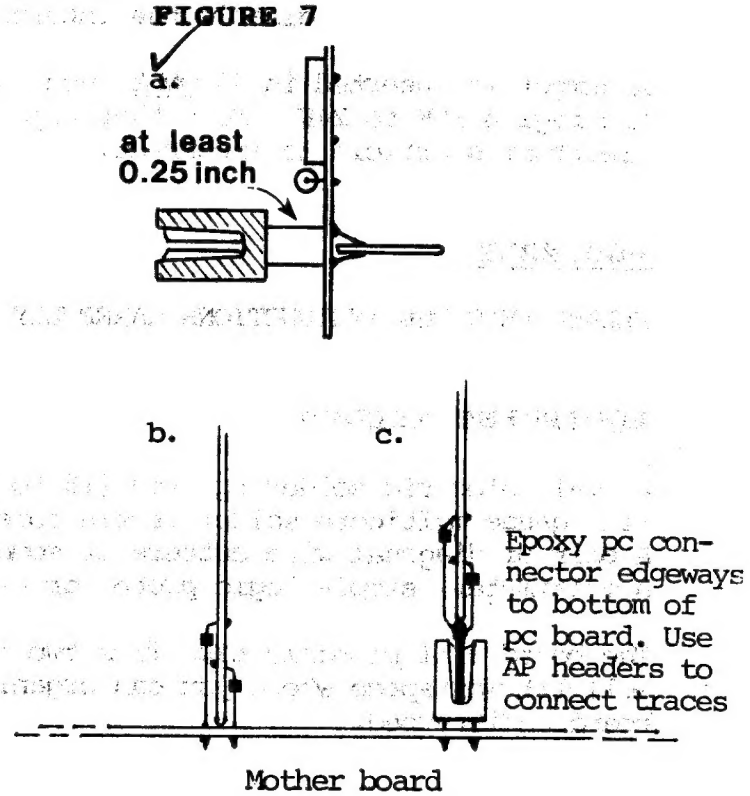


FIGURE 7



PROCEDURE

- ✓ Refer to Figure 6. Solder in place the four 24 pin IC sockets. Make sure your soldering iron is clean and hot and use a minimum amount of solder. The bond should be shiny in appearance -- not a dull grey. Cold solder joints and solder bridges between traces are the most probable causes of ultimate malfunction. Always double check that the component is in the correct place before soldering. It's sometimes very difficult to remove a component from a pc board without damage either to the board or to the component.

- ✓ Solder the 16 pin IC socket in place -- note the polarity.

- ○ Refer to Figure 7. Note that there are three ways to attach the memory board to your ZX81. If you have an expansion or mother board with the same signal arrangement you might prefer to use AP headers as shown in Figure 7b or c. Otherwise use the method shown in Figure 7a as follows. Insert the edge connector in the appropriate holes in the board. Adjust the stand-off from the board to at least Ø.25". Make sure the connector is parallel to the board and solder one pin at each end. Check the alignment again and if you're satisfied solder the remaining pins on the connector.

- ○ On the circuit side of the board, bend the wrap post pins at each end of the socket in toward each other as shown in Figure 7. Secure the edge connector board between the two pairs and check the alignment. Solder these four pins. Now bend the remaining pins down onto the connector. Make sure that each pin is in contact with the appropriate trace on the connector. Solder the pins. (Make sure that the polarizing key is in the correct slot.)

- ○ Solder in place the 1ØØ ohm resistor R1 (brown-black-brown). See note below about omitting R1 (Fifth instruction on next page).

- ○ Solder in place the four 1K resistors R2-R5 (brown-black-red).

- ○ Solder in place the six 1ØK resistors R6-R1Ø & R12 (brown-black-orange).

- ○ Solder in place the 1ØØK resistor R11 (brown-black-yellow).

- ○ Solder in place the three diodes D1 - D3. Note the polarity -- the black band toward the top of the board. Note that D3, the germanium diode, is larger than the silicon diodes.

- ✓ Solder in place the four ceramic capacitors C2 - C5. These can go in either way round. Note that C3 can be omitted -- see note on the write-protect switch.

- ○ Solder the tantalum capacitor C1 as close to the board as possible — the positive lead must go in the hole nearer the bottom of the board. (For a ZX8Ø computer, mount C1 on the circuit side.)

- ○ Solder the five NPN silicon transistors in their appropriate places. Note that the curved side of all transistors face the left. The transistor leads must be gently spread apart before mounting them in the appropriate holes. Be careful not to overheat the transistors when soldering them — let the transistor cool before going on to the next lead. Mount Q1 as close to the board as possible.

- ◯ Solder jumper J1 in the appropriate holes (see Figure 2). If you intend to vary the 8K memory block to which the board is assigned then you may wish to use sockets for the jumper. Leads cut from the diodes or resistors may be used as jumper wires.

- ◯ Jumper J2 is configured as shown in Figure 3 for 6116LP-3 CMOS RAM. Insert the two jumper wires and solder in place.

- ◯ Jumper J3 should not be inserted.

- ✓ Jumper J4 is a diode D4. Note the polarity (black band to the right). Insert and solder in place.

- ✓ Add the reset switch on the back of the board where there is more room for your finger. You may of course omit the switch and its associated resistor R1. It is recommended, however, that you use the switch to reset the CPU instead of pulling the power plug -- if the power is removed and then quickly restored the filter capacitor in the power supply, and the reset capacitor in the ZX81, may not fully discharge. This means that the Z80 CPU may accidentally write over your nonvolatile memory when it comes alive.

- ◯ Jumpers J5 and J6 are not used and should be left vacant.

- ◯ Solder the lithium cell holder on the reverse side of the board. Press the holder firmly against the board while soldering in place.

- ◯ At this stage the board should be thoroughly checked. Look at all the solder joints and recheck the placement of components. Refer to Figure 8 and connect +5V and ground to the appropriate pins of the edge connector. Check for the presence of +5 and ground at each IC socket. Pinouts of the IC's are illustrated in Figure 9. Disconnect the supply.

- ◯ Plug IC5 into its socket. If the pins are spread too wide bend them in very gently by pressing an entire row of pins on one side of the IC against a hard surface. A piece of aluminum foil on wood works well and provides a grounded surface.

- ◯ CMOS devices are in general susceptible to damage by static electricity. Before taking the CMOS memory out of its protective foam ground yourself to remove any static charge. If you need to rest it somewhere place it on a piece of aluminum foil. Never place the device on a non conducting surface. You will probably have to bend the pins inward in order to fit the IC in its socket. Do so by gently pressing a row at a time on a grounded surface as described above. Plug the memory into the first socket (Socket 0).

- ◯ Do not add the lithium cell yet.

FIGURE 8

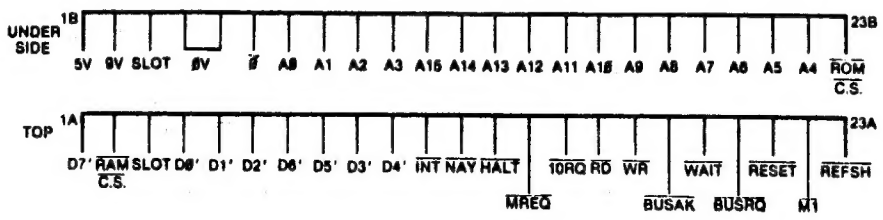
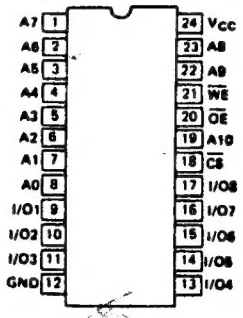
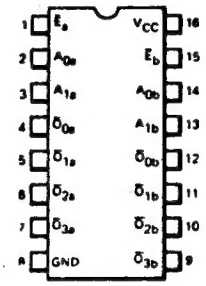


FIGURE 9



HM6116LP-3



74LS139

MEMORY CHECK

ALWAYS SWITCH OFF YOUR ZX81 BEFORE ADDING OR REMOVING THE 8K BOARD.

Its just as well to check the memory board with only one 2K CMOS RAM IC plugged in. If it works with one then the board can be fully populated and tested again.

Switch off the power and plug the board into the back of your ZX81. Do not add any other peripherals at this stage (one thing at a time). Switch on.

If the CMOS RAM board is to be used in the 8K slot from 16 to 24K then the Sinclair software can be used to test the memory. The top of system RAM is a variable 'RAMTOP' stored in memory locations 16388 and 16389 (decimal). The command PRINT (PEEK 16388 + 256*PEEK 16389)/1024-16;"K" will print the size of system RAM. Note that not all of this memory is available to the user. Some of it is used for system variables (like 'RAMTOP'), some for the display file, some for the Z80 stack, etc.

16-24K

If the CMOS RAM board is used in the 8 to 16K slot, then a simple test to check whether the board is functioning properly is to POKE 9000,N and then PRINT PEEK 9000 and see if you get the same N back again. (N is any decimal number between 0 and 255.) The address 9000 can be any number between 8192 and 10239 if the CMOS RAM chip is in Socket 0. The following BASIC program will test every address in the 2K of transparent RAM. It takes a long time (even in FAST mode -- the SLOW mode takes 4.5 minutes) and the machine language routine on the next page completes the test in a fraction of the time. This machine language routine is placed in a REM statement in system RAM -- intentionally or it would self-destruct.

BASIC PROGRAM TO CHECK EVERY MEMORY LOCATION BETWEEN 8192 AND 10239

```
 5   FAST
10   FOR N = 8192 TO 10239
20   POKE N, 1
30   NEXT N
40   LET A = 0
50   FOR N = 8192 TO 10239
60   IF PEEK N-1=0 THEN LET A=A+1
70   NEXT N
80   PRINT "CMOS MEMORY = ";A/1024;"K"
```

MACHINE LANGUAGE PROGRAM TO CHECK EVERY MEMORY LOCATION BETWEEN 8K and 16K

First enter the BASIC program:

```
10 REM 123456789012345678901
20 PRINT "TRANSPARENT CMOS MEMORY":USR 16514/1024;"K"
30 STOP
```

Now POKE in the following machine language program. You can poke the data into each location individually or you can use a short program like that on page 15.

Address	Data			Assembly language (Mnemonic)
16514	33	255	63	LD HL, (top of memory)
	1	0	0	LD BC, 00
16520	62	31		LD A, 31
	54	01		LD (HL), 01
	43			DEC HL
	188			CP H
	32			JNZ
	250			back 5 lines
	35			INC HL
	3			INC BC
16530	53			DEC (HL)
	40			JZ
	251			back 4 lines
	11			DEC BC
16534	201			RET

If you list the BASIC program now you will see that the REM statement contains characters corresponding to the data POKEd in. Now enter RUN.

BATTERY BACK-UP

At this point, if all is well, the lithium cell can be placed in its holder -- slip it gently under the positive terminal. POSITIVE SIDE UP.

When the 8K nonvolatile memory board is not in use, leave it plugged into the ZX81 -- even when the ZX81 is not switched on. Doing this effectively keeps the address and data lines to the CMOS RAM chips from floating high and ensures that the drain on the lithium cell stays below 1 microamp for a fully populated board -- an insignificant current.

An equally good technique is to plug a scrap piece of double-sided pc board into the 44-pin connector when the board is disconnected from the computer. (Short both sides of the double-sided pc board together.)

WHAT TO DO IF IT DOESN'T WORK

Remove the CMOS IC and leave in a safe place -- on aluminum foil or back in its foam. Remove IC5.

Visually check the entire board -- look for poor solder joints and solder bridges. They are the most likely causes of malfunction. The most probable place for a faulty solder joint is on the edge connector.

Make sure all components are placed correctly.

Make sure the jumpers are correctly placed.

Without the IC's plugged in, connect the board to the ZX81 again and switch on. If you do not get a cursor then there must be either a solder bridge or an intermittent contact (again probably on the edge connector). If you obtain a cursor, power down, remove the board, plug in the IC's, and try again.

Clean the edge connector in your computer with a pencil eraser. Replace the IC's in their sockets and plug the board back in securely. Retest.

If the board still does not work remove the IC's again and check the continuity of all lines from edge connector to sockets and between sockets.

If the board works alone but not with a 16K RAM pack then look for a poor solder joint between the edge connector pins and the edge connector board. Sometimes solder flux on the edge connector inhibits contact -- make sure it is clean.

All parts supplied with this kit are guaranteed. Any part found to be defective will be replaced free of charge. Any part accidentally damaged by you will be replaced at cost. (List upon request.)

With some RAM packs, notably the Sinclair 16K RAM pack, you may experience a "wobble" problem. This lack of stability in the RAM pack causes intermittent contact in the signal lines between the 8K nonvolatile memory board and the RAM pack and will often lead to a system crash. A solution is to extend a rigid base backward from the computer. Double-sided adhesive foam can be used to fix the various components of the system to this base.

Occasionally, with about 2% of all ZX81/TS1000's, you may see the characters on the screen appear to disintegrate into "chinese characters" after the machine has been on 5 or 10 seconds. This problem will often disappear if the board is fully populated or if a RAM pack is added on behind. In any event, the cure is straightforward and if you experience this problem let me know immediately.

OTHER OPTIONS

TO USE WITH A ZX80

A ZX80 computer does not have $\overline{\text{ROMCS}}$' available at the edge connector. It is necessary to make it available at 23B if the board is to be used for the transparent 8K block. Break the trace from the decoder IC6 pin 7 to the ROM pin 20 in the ZX80, bridge the break with a 680 ohm resistor, and wire a jumper from the $\overline{\text{CS}}$ pin 20 on the ROM to pin 23B on the edge connector.

MODIFICATIONS FOR 2716 EPROM'S

If the board is to be fully populated by 2716 EPROM's, resistors R2 to R12, all transistors Q1 to Q5, diodes D1 through D3, and the lithium cell should be omitted. Diode D3 should be replaced by a wire jumper and wire jumpers should also be inserted to connect the collector socket of each transistor Q2 through Q5 to the corresponding emitter socket. Jumpers J2 should be placed as shown in Figure 3. Jumper J1 should remain in position 1 for the 8 - 16K assignment and diode D4 should remain inserted in jumper J4.

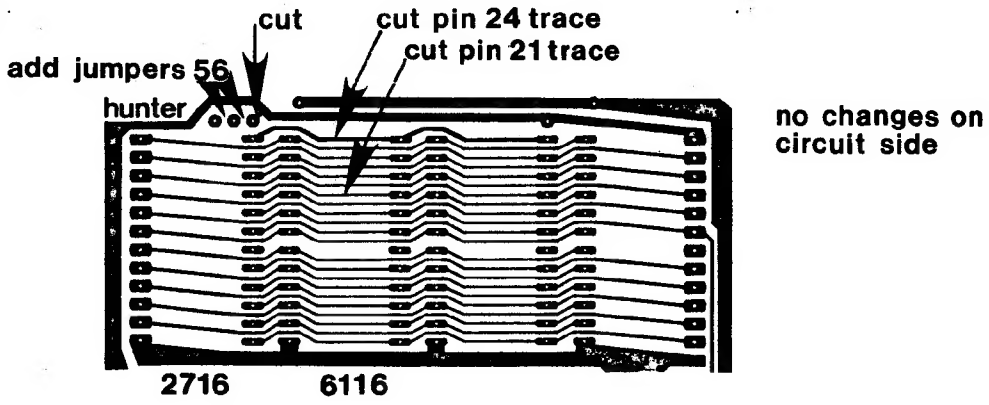
MODIFICATIONS FOR 2732 EPROM'S

The same modifications are made if the 2732 EPROM's are to be used. Sockets 1 and 3 are used for two 2732 EPROM's. Jumpers J2 are placed as shown in Figure 3. If you use the board for 2732 EPROM's note that you can fully populate the board with four 2732's (16K) and then switch, or page, between the pair in sockets Ø and 2 and the pair in sockets 1 and 3 (8K each) by switching pin 14 of the 74LS139 decoder from +5V (for sockets 1 and 3) to ground (for sockets Ø and 2). Pin 14 of the 74LS139 can be accessed via the upper left hole in J2.

MODIFICATIONS FOR A RAM/EPROM MIXTURE

HM6116 CMOS RAM's and 2716 EPROM's are pin compatible. You can replace one of the 6116's without circuit modification. For a permanent replacement use jumpers J5 and J6 to supply the full +5V to the EPROM. If J5 and J6 are inserted then both the pin 21 trace and the pin 24 trace must be cut between the 2716 and 6116LP sections. Always have the 2716 EPROM's to the left (lower address) of the 6116LP RAM's. All other traces to the pin 24 of each 2716 socket must also be cut. An example -- one 2716 and three 6116LP devices -- is illustrated in Figure 1Ø. It's also possible to have 2 2716's and 2 6116's. Note that if an EPROM is used then the transistor and its two associated resistors can be omitted for that location as described above.

FIGURE 1Ø



It is recommended that, at least for experimentation, you use two boards -- one configured for CMOS RAM and the other for EPROM. Both can be connected at the same time so long as a particular socket is occupied on only one of the boards. If you wish to do this then a kit of a second board, a wire-wrap edge-connector, a pc edge connector board, five IC sockets, a 74LS139 decoder, a tantalum capacitor, four ceramic capacitors, and a diode (all you need), may be obtained for the special price of $20 post paid if you return your initial receipt.

MODIFICATIONS FOR EEPROM'S

It is possible to read EEPROM's on the board just like the EPROM's described above. Moreover, with the addition of a device to extend the write-enable signal to 1 or 1Ø ms (depending on the EEPROM), it is possible to write into the EEPROM. Refer to an article by Joe Blagg in BYTE (February 84 page 342) or to Application note #8 by Danton Leonard of SEEQ TECHNOLOGY, 1849 Fortune Drive, San Jose, CA 95131.

MODIFICATIONS FOR USE AS SYSTEM RAM

When the board is used as system RAM the battery back-up is unnecessary and the same modifications described above for 2716 EPROM's can be made. However, if you intend to vary the use to which you put the board it is advisable to follow the assembly instructions detailed above for assignment to the transparent region. Changing over to system RAM is then done very simply by changing jumper J1 to position 2 and adding jumper J3 as illustrated in Figure 4. Note that cheaper NMOS memory IC's can be used in place of CMOS if the board is to be used as system RAM.

USING THE BOARD IN A SYSTEM WITH MORE THAN 32K

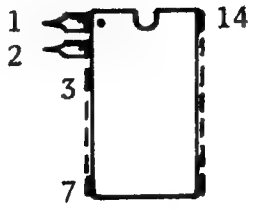
This board, like the Sinclair RAM pack, completely ignores A15 and is therefore limited to use within a 32K system. The board is enabled through $\overline{\text{MREQ}}$ only. See Figure 13. It is however possible to use the board in a system addressing more than 32K by adding additional decoding. Moreover, it is possible to use the board in any 8K slot in such a system.

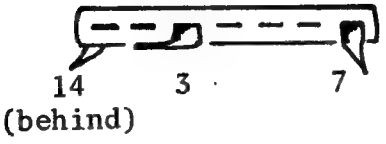
If the board is to be used as transparent memory (8-16K) in a system larger than 32K then the following procedure is suggested. In effect, A15 is combined with $\overline{\text{MREQ}}$ in enabling the 74LS139 decoder. The gate used can be an OR gate, a combination of three NAND gates, or two NOR gates, etc. The following procedure refers to the use of a 74LS32 quad 2-input OR gate.

1. Remove the 74LS139 (IC5) from its socket.

2. Very gently bend up pin 1 and then bend it over sideways:

3. Obtain a 74LS32 IC and, again very gently, straighten pins 1 and 2 so that they point directly out. Pin 3 should be bent sideways as shown below. Pins 7 and 14 should be bent sideways a little. With loss of little power (less than 1 mA current), pins 4 through 6 and pins 8 through 13 can be cut off at the body of the IC. Check the following sections in this manual before you cut off any pins.

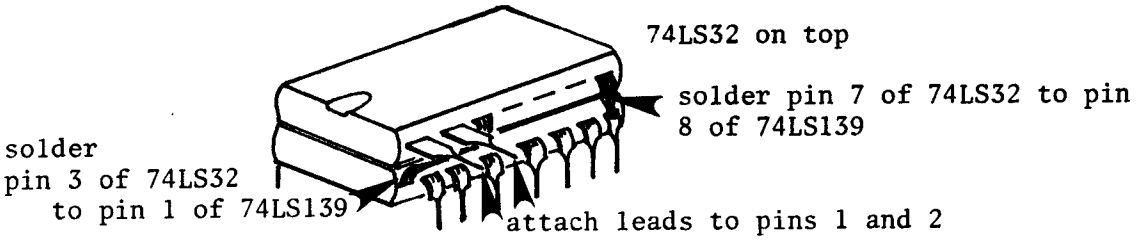




4. Mount the 74LS32 on top of the 74LS139. A thin strip of tape wrapped around their bodies will hold them together while you solder the leads. Pin 1's should be at the same end.

5. Solder together:

	74LS139		74LS32
(small amount of solder and do not overheat)	pin 16	to	pin 14
	pin 8	to	pin 7
	pin 1	to	pin 3



6. Solder flying leads (about 1" insulated) to pins 1 and 2 of the 74LS32.

7. Reinsert the dual IC package into the original socket on the board.

8. Solder the free ends of the two flying leads to A15 and $\overline{\text{MREQ}}$ at the edge connector. It doesn't matter which lead goes to which connector.

9. The board is now uniquely addressed in the 8K to 16K region and can be used in a 64K system.

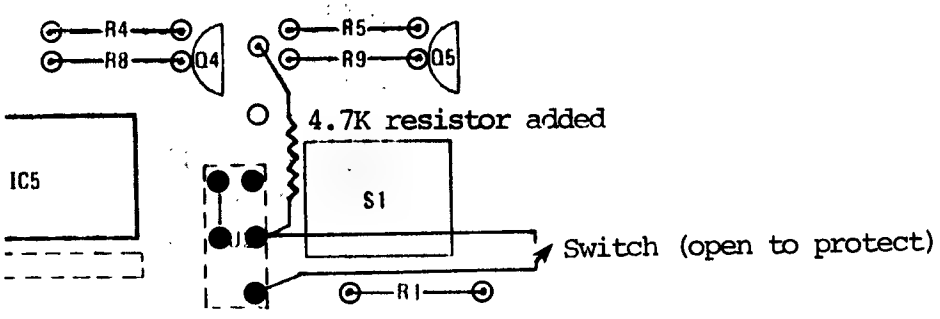
In order to use the board as RAM above 32K, it is necessary to gate (AND) $\overline{\text{MI}}$ with A15 in the decoding of the address. The article by John L. Oliger on page 5 of number 5 volume 3 (1982) of SYNTAX is recommended. The technique of mounting another IC on top of the 74LS139 can still be used.

WRITE PROTECT SWITCH

Addition of a write-protect switch to your nonvolatile memory board is worthwhile. It avoids loss of data which occasionally happens upon power-down and it can be used to prevent accidental writing over the memory when you test a machine language routine for the first time. The switch can be epoxied to the front of the NVM board opposite the reset switch.

Replace C3 with a 4.7K resistor from the top hole for C3 and the center hole of J2 as shown in Figure 11. Add the switch between the lower two holes of J2. Open the switch to disable a write operation. Note that you MUST open the switch before power-down or the current from the lithium cell will drain 100 times more quickly (back into the computer!). An alternative is to isolate the line through a transistor or one of the gates in the 74LS32.

FIGURE 11

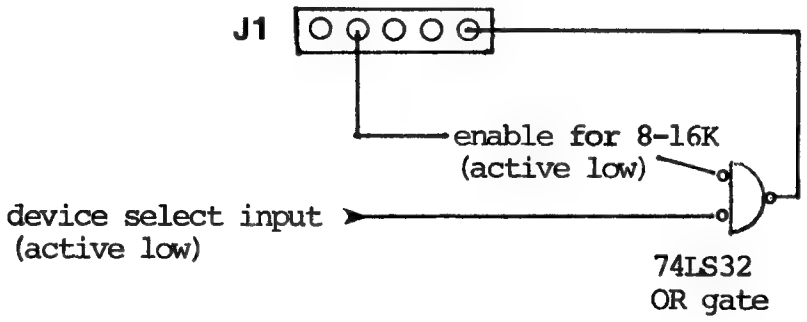


PAGING

With the proliferation of devices using the 8-16K region it would be very useful to be able to have all the devices plugged in at one time but to enable only the one you wished to use at a particular instance. In such a system, each device would be addressed by a unique enable signal. This signal may be memory-mapped (the BASIC POKE command may then be used) or it may be generated in the I/O space (the machine language OUT command must be used because the SINCLAIR BASIC has no IN or OUT command). In order not to load down the data, address, and control lines of the ZX81 it is advisable to buffer them. See for example the diagram on page 37 of the Winter 1982 issue of SQ (Syntax Quarterly).

Each peripheral device then should have a device-enable input. One can be made for the 8K nonvolatile memory board using a gate in the 74LS32 quad OR IC as shown in Figure 12. The best way to disable the NVM board is shown in Figure 12. It has the advantage of not interfering with the $\overline{\text{ROMCS}}$' signal. Jumper J1 is replaced. Several boards can be attached simultaneously using this method.

FIGURE 12

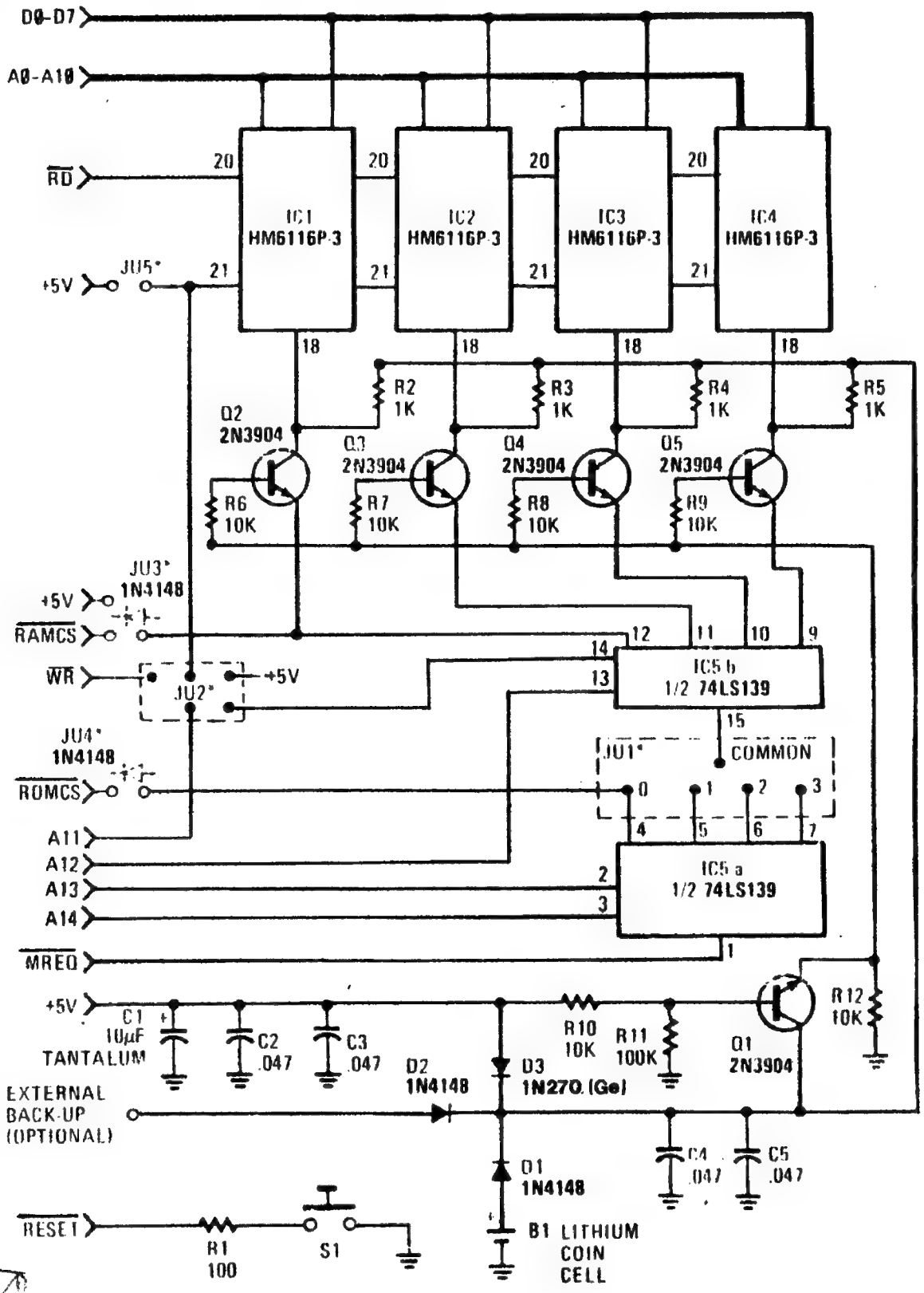


A manual switch may be substituted for the device select signal and the OR gate -- in this case tie the output of the switch to +5V via 2 to 5K.

PARTS LIST

1	Main pc board
1	Pc board edge connector
1	44 pin wire wrap edge connector
4	24 pin sockets
1	16 pin socket
1	momentary push button switch (Panasonic EVQ-P1R04K)
1	lithium cell holder (Memory Protection Devices BH906)
1	lithium 3V cell (Panasonic BR2325; Sanyo CR2032; etc.)
1	HM6116LP-3 150nsec low power 2K CMOS RAM (IC1)
1	74LS139 decoder (IC5)
3	1N4148 silicon diodes (D1-D2, D4)
1	1N270 germanium diode (D3)
5	NPN general purpose silicon transistors (2N3904) (Q1-Q5)
1	100 ohm resistor (R1; brown black brown)
4	1K resistors (R2-R5; brown black red)
1	4.7K resistor (for write-protect switch; yellow violet red)
6	10K resistors (R6-R10, R12; brown black orange)
1	100K resistor (R11; brown black yellow)
1	10uF 16V tantalum capacitor (C1)
4	0.047uF bypass capacitors (C2-C5)
1	instruction manual

FIGURE 13 SCHEMATIC DIAGRAM



A SAMPLE DISPLAY ROUTINE

The display file in the Sinclair system does not occupy a fixed location but moves around as the program length changes (see page 128 in the operating manual). If you have more than 3.25K of system RAM, the graphics area of the display file normally consists of 704 characters (22 lines of 32 characters each). In addition there is a code 118 at the beginning of the display file, and a code 118 at the end of each line. Also note that there are two more lines below the graphics area.

The first routine listed below stores all characters (704) in the graphics area of the screen in a predetermined memory location. In this example the location is the block between 9000 and 9703 decimal (inclusive). The second routine loads the data back into the display file. The third routine fills the stored display file with a defined character (poked into 8236).

You can enter the machine code using the following BASIC program:

```
100   INPUT A
110   INPUT D        When prompted, enter the address of the first byte
120   POKE A,D       (8192) and then on the next prompt enter the data (42).
130   SCROLL         Continue entering the data until the end of listing 3.
140   PRINT A,D      To quit the program enter an L for example.
150   LET A=A+1
160   GOTO 110
```

After the machine language programs in listings 1, 2, and 3, have been entered, type NEW to clear the system and then try the BASIC program:

```
10   FOR N = 1 TO 704
20   PRINT "■";
30   NEXT N
40   RAND USR 8192
```

Now switch off your computer. Switch it back on and, with more than 3.25K of memory (use your RAM pack), enter RAND USR 8210. Your display should reappear. If you don't have a RAM pack you must first expand the display file by filling the screen with a character or by poking an artificially high value into the memory location for the variable "RAMTOP". For example, without a RAM pack attached, first enter the command POKE 16389,80 and then enter the command RAND USR 8210. Now try the program (in SLOW mode):

```
10   FOR N = 1 TO 63
20   POKE 8236,N
30   RAND USR 8229 + USR 8210
40   NEXT N
```

This program will give you an idea of how fast you can change the entire display. One advantage of having a secondary display file mapped in memory is that it allows you to make changes to the file over a period of time and then display all the changes simultaneously on the screen. This allows much smoother graphics.

LISTING 1 SCREEN DUMP

Address	Data			Mnemonic
8192	42	12	64	LD HL (start of display file)
8195	17	40	35	LD DE (storage location starting address; 9000 decimal)
8198	62	22		LD A (number of lines in graphics area)
8200	35			INC HL
8201	1	32	0	LD BC (number of characters on a line)
8204	237			LDIR
8205	176			
8206	61			DEC A
8207	32			JNZ
8208	247			back 8 lines
8209	201			RETURN

LISTING 2 RELOAD DISPLAY FILE

Address	Data				Mnemonic
8210	237	91	12	64	LD DE (start of display file)
8214	33	40	35		LD HL (storage location starting address; 9000 decimal)
8217	62				LD A (number of lines in graphics area)
8218	22				
8219	19				INC DE
8220	1	32	0		LD BC (number of characters on a line)
8223	237				LDIR
8224	176				
8225	61				DEC A
8226	32				JNZ
8227	247				back 8 lines
8228	201				RETURN

LISTING 3 FILL DUMPED DISPLAY FILE WITH CHARACTER N

Address	Data			Mnemonic
8229	33	40	35	LD HL (storage location starting address; 9000 decimal)
8232	1	192	2	LD BC (number of characters in display; 704)
8235	54			LD (HL) (code for character to be displayed)
8236	128			other characters may be poked into this address
8237	35			INC HL
8238	11			DEC BC
8239	120			LD A,B
8240	177			OR C
8241	32			JNZ
8242	248			back 7 lines
8243	201			RETURN

SAVE YOUR MACHINE LANGUAGE ROUTINES ON TAPE

Occasionally, by accident, it is possible to clear the entire 8K of transparent RAM or to fill it with garbage. Reentering even 1K of machine code can be extremely tiresome and it is advisable to store the contents of the CMOS RAM on tape periodically as a back-up. (Sooner or later the lithium cell will fail). The system SAVE command does not save any programs stored in the transparent region so the contents of the 8-16K block must be moved up into the system RAM and then SAVEd. This can be done as follows:

1. Enter the program:

```
10 REM 12345678901234567890123456789012345
12 INPUT N
14 POKE 16516, INT (N/256)
16 POKE 16515, N-256*INT (N/256)
18 RAND USR 16514
20 REM
```

2. Enter the machine code in listing 4 in the REM statement on line 10. If you use a BASIC program like that on page 15 to enter the machine code remember to delete it (or place it before the last REM). RUN. When prompted, enter the amount of memory you want to save (N) -- usually 2048 (2K) or 8192 (8K). The program takes less than a second for 8K. When you now LIST you will see that the REM statement on line 20 contains N asterisks.

3. Delete lines 10 through 18 and enter line 10 again:

```
10 REM 123456789012345678901234
```

4. Enter the machine code in listing 5. These routines assume that the size of the block to be transferred is 2K. For 8K change lines 16522 and 16534 to 32. Again, you can use a BASIC program to enter the machine code (for example at line 100) but do not attempt to delete or insert any line immediately following the REM statement on line 20.

5. You now have a program consisting of two REM statements:

   10 REM (with the machine language in listing 5)
   20 REM (with 2048 or 8192 asterisks)

   Save this program on tape for future memory dumps.

6. Enter RAND USR 16514. This transfers the specified block of system-transparent memory up into the system RAM. If you LIST now, you may see some new lines with odd-looking line numbers rather than a continuous REM statement as expected. This will occur if you have any code 118's in the machine code transferred up -- the code 118 is recognized by the SINCLAIR BASIC as the end of the REM statement. This does not matter provided that you do not try to add new lines, or delete lines, etc., but go straight on to the next step (step 7).

7. SAVE "name" on tape.

8. To reload your routines into the 8-16K block, reload from tape, enter RAND USR 16526, and then enter NEW to clear the system.

LISTING 4 GENERATE REM STATEMENT

Address	Data			Mnemonic
16514	1	Ø	Ø	LD BC (size of REM statement)
16517	3			INC BC
	3			INC BC (add 2 for REM and CODE 118)
16519	42	12	64	LD HL (start of display file)
16522	43			DEC HL
	43			DEC HL
	43			DEC HL
	43			DEC HL (point to size of REM statement)
16526	113			LD (HL), C
	35			INC HL
	112			LD (HL), B (load size of REM statement)
	35			INC HL
16530	35			INC HL (point to address for new character)
	11			DEC BC (decrease BC to number of characters
	11			DEC BC to be entered)
	197			PUSH BC
	229			PUSH HL
16535	205	158	9	CALL (move everything up to make room)
16538	225			POP HL
16539	193			POP BC
16540	54	23		LD (HL), (character *)
16542	35			INC HL
	11			DEC BC
	120			LD A, B
	177			OR C
	32			JNZ
	248			(back 7 lines)
16548	201			RETURN

LISTING 5 DUMP OR LOAD TRANSPARENT MEMORY

Address	Data			Mnemonic
16514	33	Ø	32	LD HL (start of transparent RAM; 8192)
16517	17	160	64	LD DE (start of area in REM statement; 16544)
16520	1	Ø	8	LB BC (size of block to be transferred; 2048)
16523	237			LDIR
	176			
16525	201			RETURN
16526	33	160	64	LD HL (start of area in REM statement; 16544)
16529	17	Ø	32	LD DE (start of transparent RAM; 8192)
16532	1	Ø	8	LB BC (size of block to be transferred; 2048)
16535	237			LDIR
	176			
16537	201			RETURN

MERGING BASIC PROGRAMS

Having an area of memory transparent to the BASIC system allows you to merge BASIC programs and/or data easily. The following procedure is suggested:

1. Load one program from tape if not already resident in your system.
2. Move the program down to the transparent region (USR 8244).
3. Load the second program from tape.
4. Generate space in the system RAM for the merge.
5. Move the first program back up into the system RAM (USR 8269).

LISTING 6 DUMP TO TRANSPARENT MEMORY

Address	Data			Mnemonic
8244	205	35	15	CALL fast mode
8247	17	125	64	LD DE (start of program area; 16509)
8250	42	12	64	LD HL (start of display file)
8253	183			OR A (clear carry flag)
8254	237	82		SBC HL, DE
8256	68	77		LD B, H and LD C, L (store difference in BC)
8258	33	40	35	LD HL (start of storage area; 9000)
8261	113			LD (HL), C (store C @ 9000)
8262	35			INC HL
8263	112			LD (HL), B (store B @ 9001)
8264	35			INC HL
8265	235			EX DE, HL
8266	237	176		LDIR
8268	201			RETURN

LISTING 7 GENERATE SPACE AND RELOAD PROGRAM

Address	Data				Mnemonic
8269	205	35	15		CALL fast mode
8272	237	75	40	35	LD BC (size of program)
8276	42	12	64		LD HL (start of display file)
8279	197	229			PUSH BC; PUSH HL
8281	43				DEC HL
8282	205	158	9		CALL routine at 2462 decimal
8285	209	193			POP DE; POP BC (note exchange)
8287	33	42	35		LD HL (start of storage area; 9002)
8290	237	176			LDIR
8292	201				RETURN

Enter the two routines in listings 6 and 7. Load the first BASIC program to be merged -- note that this program will eventually be at the higher address (higher line numbers). Enter RAND USR 8244. This routine moves the BASIC program down into the transparent RAM.

Clear the system and enter or load the second BASIC program. Do not duplicate line numbers taken by the first BASIC program — make all the line numbers in this BASIC program less than the lowest line number in the first BASIC program dumped above. Enter:

IF USR 8269=0 THEN STOP (the STOP prevents automatic execution of the program)

This merge routine can be used to store BASIC programs in the 8-16K nonvolatile region and retrieve them. The first BASIC program retrieved could be an index which in turn could retrieve other programs by poking appropriate addresses into memory locations 8274, 8275, 8288, and 8289. A very similar program can be written for moving, storing, and merging variables. Let me know if you need a copy.

The following BASIC program is useful in renumbering lines:

```
9900   REM LINE RENUMBER
9905   PRINT "NOTE GOTO AND GOSUB ADDRESSES"
9910   PRINT
9915   PRINT "ENTER NUMBER FIRST LINE WILL BE"
9920   INPUT F
9925   PRINT
9930   PRINT "ENTER LINE INCREMENT"
9935   INPUT I
9940   PRINT
9945   LET N = 16509
9950   POKE N, INT (F/256)
9955   POKE N+1, F - 256*INT (F/256)
9960   REM POINT TO NEXT LINE
9965   LET N = N + PEEK (N+2) + 256*PEEK (N+3) + 4
9970   IF 256*PEEK N + PEEK (N+1) = 9900 THEN GOTO 9990
9975   LET F = F+I
9980   GOTO 9950
9990   PRINT "LAST LINE IS ";F;
```

A machine code version of a similar renumber program has been published in SYNTAX, volume 3, #6 (June 1982) page 4 in an article by Mike Mullen.

AMOUNT OF FREE SPACE IN MEMORY

The amount of free space in memory is the "spare" memory between the top of calculator stack and the bottom of the machine stack. See page 128 in the ZX81 manual. The top of the calculator stack is stored as the ZX81 system variable "STKEND" and the bottom of the machine stack is always pointed to by the Z80 register sp (stack pointer). The following routine simply subtracts the two. To determine the amount of free memory in your system, enter the command PRINT USR 8297. The result will be the number of free bytes.

LISTING 8 AMOUNT OF FREE SPACE

Address	Data				Mnemonic	
8297	33	0	0		LD HL, 00	Clear HL
8300	57				ADD HL, SP	Move SP to HL
8301	237	91	28	64	LD DE (Stkend)	
8305	237	82			SBC HL, DE	Carry flag already cleared
8307	68	77			LD B, H and LD C, L	
8309	201				RETURN	